# Virtual memory management [Assignment-6]

**Objective:**
Simulate and analyze the performance of page replacement techniques: LRU, clock, and FIFO by varying the page size.
With pure-demand paging technique *with and without pre-paging feature*.

Application describes the process of the management of the virtual memory in all cases of page faults and the replacement of pages with some widely used algorithms. Considering that the main memory size of size 500 and by varying the page size from 1,2,4,8,16,32,.... and initiating 10 processes with a given memory references and program address limits.

**Algorithms :**
Once the frame size is decided, the pages starts loading into the memory according the following algorithms.
a. ***Local replacement***.
1. LRU : Here , page table have an extra attribute based on its usage. A variable for every process called RecentUse[p] is given to every page of process p that is being placed in the memory or that is being used for every new page, the value RecentUse[p] gets incremented.

If all the space in the memory is filled according to the limits then for replacing , we go for -> the page with least value of RecentUse[p] is replaced by the new page.

2.FIFO : Every page during its entry gets an arrival number and it is stored in the page table and this arrival number is a local value ie., every process has its own arrival number and increments by one for every new page from its process to load.

If all the space in the memory is filled according to the limits then for replacing , we go for -> the page with least arrival number in its process is replaced by the new page.

3. Clock : Here page table have an extra attribute called as reference bit ( which could be 0 or 1 ) . When a page gets freshly placed in the memory ( either by replacing some other page or in the empty space in memory), it is given with ‘1’ as reference bit.

Whenever a new page tries to replace any of the page, it starts checking for the page with reference bit ‘0’, if it come accross with the page with reference bit ‘1’, it changes the bit from ‘1’ to ‘0’ and this search process is cyclic, ie., like a clock.
The new pages checks / passes only over the pages of it’s process only.

b. ***Global replacement.***
1. LRU : Here , page table have an extra attribute based on its usage. A global variable called RecentUse is given to every page that is being placed in the memory or that is being used for every new page, the value RecentUse gets incremented.

If all the space in the memory is filled according to the limits then for replacing , we go for  the page with least value of RecentUse is replaced by the new page.

2. FIFO :  Every page during its entry gets an arrival number and it is stored in the page table and this arrival number is a global value and increments by one for every new page to load, irrespective of the process.

If all the space in the memory is filled according to the limits then for replacing , we go for  the page with least arrival number is replaced by the new page.

3.Clock : Here page table have an extra attribute called as reference bit ( which could be 0 or 1 ) . When a page gets freshly placed in the memory ( either by replacing some other page or in the empty space in memory), it is given with ‘1’ as reference bit.

Whenever a new page tries to replace any of the page, it starts checking for the page with reference bit ‘0’, if it come accross with the page with reference bit ‘1’, it changes the bit from ‘1’ to ‘0’ and this search process is cyclic, ie., like a clock. And the process is global, the new page moves over all the frames irrespective of their process.

**Statistics**:

The statistics observed are the number of page faults and replacements occurred for every algorithm used.

| Page Size . | FIFO - SIMPLE - L | CLOCK - SIMPLE - L | LRU - SIMPLE - L | FIFO - PREPAGING - L | CLK - PREPAGING - L | LRU - PREPAGING - L | FIFO - SIMPLE - G | CLOCK - SIMPLE - G | LRU - SIMPLE - G | FIFO - PREPAGING - G | CLK - PREPAGING - G | LRU - PREPAGING – G |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 12438.3 | 11982.7 | 11983.7 | 6746.5 | 4559.8 | 6396.9 | 11841.4 | 11703.5 | 11693.9 | 6413.1 | 6323.7 | 6319.3 |
| 2 | 9567.2 | 8950.6 | 8947.8 | 6582.6 | 3713 | 6048.7 | 9316.5 | 8747.4 | 8759.9 | 6312.9 | 6005.7 | 5960.8 |
| 4 | 8305.4 | 7421.6 | 7418.1 | 7181.6 | 3184 | 5987.5 | 8029.1 | 7255.9 | 7254.7 | 6831.9 | 5962.7 | 5899.6 |
| 8 | 8125.6 | 6674.2 | 6650.5 | 8680.6 | 4065.8 | 6647.2 | 7745.1 | 6526.6 | 6526 | 8210.3 | 6377.8 | 5935.8 |
| 16 | 8817.2 | 6620.8 | 6463.8 | 10333.7 | 6422.8 | 15634.7 | 8426.7 | 6176.3 | 6185.1 | 12036.3 | 8400 | 6609.6 |
| 32 | 11555.5 | 10505.1 | 10440.9 | 10294.8 | 10294.8 | 29222 | 12514.9 | 12279.1 | 9487.4 | 11645.8 | 11645 | 12071.3 |